UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P O Box 1450
Alexandria, Virginia 22313-1450
www uspto gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/676,634	10/01/2003	Luis M. Gomes	5150-82801	7873

7590 10/06/2009

Jeffrey C. Hood
Meyertons, Hood, Kivlin, Kowert & Goetzel PC
P.O. Box 398
Austin, TX 78767

EXAMINER	
AUGUSTINE, NICHOLAS	
ART UNIT	**PAPER NUMBER**
2179	
MAIL DATE	**DELIVERY MODE**
10/06/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

**BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES**

Application Number: 10/676,634
Filing Date: October 01, 2003
Appellant(s): GOMES ET AL.

Jeffrey C. Hood
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 06/23/2009 appealing from the Office action mailed 04/10/2009.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is incorrect. A correct statement of the status of the claims is as follows:

This appeal involves claims 1 and 6-23. Claims 1 and 6-23 are pending in the case. Claims 1 and 6-23 stand rejected under 35 U.S.C. 103(a) and are the subject of this appeal. A copy of claims 1 and 6-23, incorporating entered amendments, as on appeal, is included in the Claims Appendix hereto.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

2004/0001106	DEUTSCHER	11-2004
2004/0221262	HAMPAPURAM	11-2004

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

3. Claims 1 and 6-23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Deutscher et al. (2004/0001106), herein referred to as Deutscher in view of Hampapuram et al. (US 2004/0221262 A1), herein referred to as Hampapuram.

As for independent claims 1,18-21, Deutscher teaches a memory medium which stores program instructions implementing a graphical user interface (GUI) for a program and corresponding method and system for debugging a program, wherein, during execution of the program, the program instructions are executable by a processor to perform: displaying source code for the program on a display during execution of the program, wherein the executing program was compiled from the source code (par.137, 155, 200-201; figures 13,17 and 26B; wherein depicted in figure 17 is the displaying of program code in the pop-up window during program execution); receiving first user input hovering a mouse cursor over an expression in the source code during execution of the program (par. 180, 183 (a hover event is initiated to present a pop-up window; and par. 200-201); in response to said hovering the mouse cursor over the expression, displaying a GUI element proximate to the expression, wherein the GUI element includes a value of the expression; receiving second user input to the GUI element modifying the displayed value, thereby specifying a new value for the expression; and setting the expression in the program to the new value in response to the second user input, wherein the program continues execution in accordance with the new value of the expression (paragraph 143; wherein Deutscher explains how the user can double click an expression in the browser to display a pop-up edit box in a proximate location to the mouse as depicted in figures 13 and 17).

Deutscher does not specifically make the connection of figure 13 and 24A pop-up windows that hovering with a mouse can be used for the pop-up window of figure 13 which is used for pop-up window 24A, only that Deutscher gives an example of a mouse interaction trait being that of "double clicking" for pop-up window in figure 13. It would have been obvious to one of ordinary skill in the art at the time of the invention was made in include the functionality of hovering a mouse over the expression as well as double clicking, this is true because hovering the mouse and double clicking the mouse are very well known common mouse events in computer programs and because *Deutscher gives only for example and does not limit the system to only double clicking event from the mouse* for the pop-up window in figure 13 gives probable cause for an obvious variant of any mouse events such as hovering featured in figure 24A. Deutscher also provides to the user the ability to use a standard hover maneuver to show a pop-up window (par.180, and figures 23-24C), although Deutscher is talking about a different pop-up window than shown in figure 13 the connection between the two pop-up windows (figure 13 and 24A) and the method of obtaining visual presentation of the two pop-up windows (double clicking and hovering respectively) one of ordinary skill in the art would make the determination that both figures 13 and 24A are pop-up windows as explained by Deutscher that use two different methods of presenting themselves, double clicking and hovering, and that figure 13 could be displayed by the method used to present figure 24A and vice-versa, hence because they are both pop-up windows and using one mouse event or another available by Deutscher's system would yield the predictable result to have a user hover over an area of interest to present the pop-up window shown in figure 13. Furthermore Hampapuram teaches the user using a mouse to hover over an area of interest to display a pop-up window called a "tool tip" (400) in paragraph 44. Deutscher does not specifically teach that the user definable information is source code (programming language) from an executable program, however Hampapuram teaches the use of user definable information being that of source code from an executable program (par.7, 20 and 22).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include Hampapuram into Deutscher, this is true because to one of ordinary skill in the art would recognize the program being used in the system of Deutscher does not have to be program specific for the functionality of a pop-up control and that the pop-up control could work in any program environment (e.g. debugger) and as suggested by Hampapuram to display the code in a variety of ways (see par. 9, tooltip, popup, pane, window, etc...). Thus the combination of Hampapuram into Deutscher would yield the predictable result of having a control pop-up window which is initiated by hovering with the mouse courser over an area of interest by the user in such that the user is able to input data into the pop-up window upon presentation of pop-up window by the system.

Deutscher does not specifically mention that the program being used in the system is a debugger program. However in the same field of endeavor Hampapuram teaches a debugging program for displaying source code for the program on a display during execution of the program (figure 3; paragraph 20). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include Hampapuram into Deutscher, this is true because to one of ordinary skill in the art would recognize the program being used in the system of Deutscher does not have to be program specific for the functionality of a pop-up control and that the pop-up control could work in any program (e.g. debugger). Also Deutscher system is related to a debugger in the sense that it is a developer (author) software used for creating program presentations wherein the user can edit a program then preview, stop the preview, edit and preview again with this software (figure 11; paragraphs 137-140).

As for dependent claim 6, Deutscher teaches the memory medium of claim 1, wherein the GUI element is context sensitive (figure 17).

As for dependent claim 7, Deutscher teaches the memory medium of claim 6, wherein the GUI element comprises a control corresponding to a data type of the expression, and wherein the data type of the expression comprises at least one of: a string data type; a character data type; a numeric data type; a Boolean data type; and an array data type (figure 13 and 17).

As for dependent claim 8, Deutscher teaches the memory medium of claim 6, wherein the GUI element is operable to display the value of the expression in a specified format; wherein if the expression comprises integer data, the specified format comprises one or more of: decimal; hexadecimal; octal; binary; and ASCII; and wherein if the expression comprises single or double precision, the specified format comprises one or more of: floating point; and scientific notation (figure 8 and 17).

As for dependent claim 9, Deutscher teaches the memory medium of claim 8, wherein the specified format is specified via a second GUI element in the GUI (figure 17).

As for dependent claim 10, Deutscher teaches the memory medium of claim 1, wherein the GUI element comprises: a first portion, operable to display the value of the expression, wherein the first portion is further operable to receive the second user input modifying the value; and a second portion, operable to display non-editable information related to the expression (par.137, 155, 200-201; figures 13, 17 and 26B).

As for dependent claim 11, Deutscher teaches the memory medium of claim 10, wherein the second portion comprises a text indicator, operable to display text (figure 17).

As for dependent claim 12, Deutscher teaches the memory medium of claim 10, wherein the first portion is further operable to graphically indicate that the value is editable (figure 15).

As for dependent claim 13, Deutscher teaches the memory medium of claim 1, wherein the expression comprises a variable (figures 8, 13, 15 and 17).

As for dependent claim 14, Deutscher teaches the memory medium of claim 1, wherein the expression comprises a syntactic expression comprising one or more of:
one or more variables; one or more constants; one or more macros; and
one or more operators (figure 15 and 17). Deutscher does not specifically teach that the user definable information is source code (programming language) from an executable program, however Hampapuram teaches the use of user definable information being that of source code from an executable program (par.7, 20 and 22).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include Hampapuram into Deutscher, this is true because to one of ordinary skill in the art would recognize the program being used in the system of Deutscher does not have to be program specific for the functionality of a pop-up control and that the pop-up control could work in any program environment (e.g. debugger) and as suggested by Hampapuram to display the code in a variety of ways (see par. 9, tooltip, popup, pane, window, etc...). Thus the combination of Hampapuram into

Deutscher would yield the predictable result of having a control pop-up window which is initiated by hovering with the mouse courser over an area of interest by the user in such that the user is able to input data into the pop-up window upon presentation of pop-up window by the system.

As for dependent claim 15, Deutscher teaches the memory medium of claim 1, wherein the execution of the program is in debugging mode (note the analysis of claim 1; debugging program taught by Hampapuram). Deutscher does not specifically teach that the user definable information is source code (programming language) from an executable program, however Hampapuram teaches the use of user definable information being that of source code from an executable program (par.7, 20 and 22).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include Hampapuram into Deutscher, this is true because to one of ordinary skill in the art would recognize the program being used in the system of Deutscher does not have to be program specific for the functionality of a pop-up control and that the pop-up control could work in any program environment (e.g. debugger) and as suggested by Hampapuram to display the code in a variety of ways (see par. 9, tooltip, popup, pane, window, etc...). Thus the combination of Hampapuram into Deutscher would yield the predictable result of having a control pop-up window which is initiated by hovering with the mouse courser over an area of interest by the user in such that the user is able to input data into the pop-up window upon presentation of pop-up window by the system. *Deutscher does not specifically mention that the program being used in the system is a debugger program.* However in the same field of endeavor Hampapuram teaches a debugging program for displaying source code for the program on a display during execution of the program (figure 3; paragraph 20). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include Hampapuram into Deutscher, this is true because to one of ordinary skill in the art would recognize the program being used in the system of Deutscher does not have to be program specific for the functionality of a pop-up control and that the pop-up control could work in any program (e.g. debugger). Also Deutscher system is related to a debugger in the sense that it is a developer (author) software used for creating program presentations wherein the user can edit a program then preview, stop the preview, edit and preview again with this software (figure 11; paragraphs 137-140).

As for dependent claim 16, Deutscher teaches the memory medium of claim 1, wherein the program instructions are further executable to perform: evaluating the expression to determine the value of the expression (note the analysis of claim 1; debugging program taught by Hampapuram). Deutscher does not specifically teach that the user definable information is source code (programming language) from an executable program, however Hampapuram teaches the use of user definable information being that of source code from an executable program (par.7, 20 and 22).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include Hampapuram into Deutscher, this is true because to one of ordinary skill in the art would recognize the program being used in the system of Deutscher does not have to be program specific for the functionality of a pop-up control and that the pop-up control could work in any program environment (e.g. debugger) and as suggested by Hampapuram to display the code in a variety of ways (see par. 9, tooltip, popup, pane, window, etc...). Thus the combination of Hampapuram into Deutscher would yield the predictable result of having a control pop-up window which is initiated by hovering with the mouse courser over an area of interest by the user in such that the user is able to input data into the pop-up window upon presentation of pop-up window by the system. *Deutscher does not specifically mention that the program being used in the system is a debugger program.* However in the same field of endeavor Hampapuram teaches a debugging program for displaying source code for the program on a display during execution of the program (figure 3; paragraph 20). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include Hampapuram into Deutscher, this is true because to one of ordinary skill in the art would recognize the program being used in the system of Deutscher does not have to be program specific for the functionality of a pop-up control and that the pop-up control could work in any program (e.g. debugger). Also Deutscher system is related to a debugger in the sense that it is a developer (author) software used for creating program presentations wherein the user can edit a program then preview, stop the preview, edit and preview again with this software (figure 11; paragraphs 137-140).

As for dependent claim 17, Deutscher teaches the memory medium of claim 1, wherein the program instructions are further executable to perform: dismissing the GUI element based on one or more of: third user input, indicating dismissal of the GUI element; and elapse of a specified time period (paragraph 143).

As for dependent claim 22, Deutscher teaches the memory medium of claim 21, wherein the window is substantially just large enough to display the value of the indicated expression (note the analysis of claim 1; debugging program taught by Hampapuram wherein Hampapuram depicts a tooltip).

As for dependent claim 23, Deutscher teaches the memory medium of claim 21, wherein the window is further operable to display the indicated expression, and wherein the program instructions are further executable to perform: displaying the indicated expression with the value in the window, wherein the window does not include visible boundaries demarcating the displayed expression and value, wherein the window is substantially just large enough to display the indicated expression and the value of the indicated expression (note the analysis of claim 1; debugging program taught by Hampapuram wherein Hampapuram depicts a tooltip).

(10) Response to Argument

Beginning on page 7 of Appellant's brief (herein after "Brief"), Appellant argues specific issues, which are accordingly addressed below.

A1. As for claims 1, 10-12, 17-21, Appellant argues on pages 7-14, that Deutscher and Hampapuram does not teach "displaying source code for the program on a display during execution of the program, wherein the executing program was compiled from the source code, specifically Appellant argues that Deutscher does not teach the source code; which the source code is used in conjunction with the other limitation of the claims.

R1. Examiner does not agree with Appellant, Deutscher does not specifically teach the exact term "source code" but instead teaches a script grind comprised of script command data, script types, script parameters, UNICODE, as well as other data, (here referred to as "script grid"); as detailed in paragraphs 137-138 and 141 of Deutscher. Each the script grid and source code performs the same functionality, wherein the source code is read by the computer to perform the final outcome of the presentation, a set of instructions read by a computer to perform a function. Script grid is read by the computer to perform the final outcome of the presentation, also described as a set of instructions read by a computer to perform a function. Each may be written by a user and composed/edited but essentially they are both provided to provide instructions to a computing processor to perform a function as detailed by the instructions as defined by a user/ programmer.

Further Deutscher provides that the script grid is partially made up of transcription's that can be added and are part of the transcription grid, featured in figure 17 is the editing interface element that features within the text box what is commonly know was a programming language where the user is editing a programming language, although Deutscher is silent on the description of the figure 17 stating the user is editing a “programming language” as it is depicted in figure 17 (par.137, 155 and 180; figure 13 and 17). Further Deutscher shows that the user can input into the popup window during execution of the program to edit the script grid and transcription grid (par. 200-201).

Although Deutscher does not specifically use the term “source code”, with these findings presented above the Examiner concluded that Deutscher anticipates the teaches of source code through the teachings of a script grid as described in paragraph 137-138,141 and the visual depiction of the editing of a transcription grid which shows a programming language being edited in figure 17.

However at an attempt to advanced prosecution the Examiner presented a 103 analysis of how Deutscher does not specifically teach that the source code was not compiled code. Hampapuram is now relied upon to show that the combination of references has compiled source code wherein Deutscher is used to show the graphical user interface element (tooltip, window or pop-up) used to edit a user definable set of information and wherein Hampapuram is relied upon to show the obvious variant that the user definable set of information can be compiled source code as shown by Hampapuram.

Hampapuram’s Analysis:

Deutscher does not specifically teach that the user definable information is source code (programming language) from an executable program, however Hampapuram teaches the use of user definable information being that of source code from an executable program (par.7, 20 and 22).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include Hampapuram into Deutscher, this is true because to one of ordinary skill in the art would recognize the program being used in the system of Deutscher does not have to be program specific for the functionality of a pop-up control and that the pop-up control could work in any program environment (e.g. debugger) and as suggested by Hampapuram to display the code in a variety of ways (see par. 9, tooltip, popup, pane, window, etc...). Thus the combination of Hampapuram into Deutscher would yield the predictable result of having a control pop-up window which is initiated by hovering with the mouse courser over an area of interest by the user in such that the user is able to input data into the pop-up window upon presentation of pop-up window by the system.

Deutscher does not specifically mention that the program being used in the system is a debugger program. However in the same field of endeavor Hampapuram teaches a debugging program for displaying source code for the program on a display during execution of the program (figure 3; paragraph 20). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include Hampapuram into Deutscher, this is true because to one of ordinary skill in the art would recognize the program being used in the system of Deutscher does not have to be program specific for the functionality of a pop-up control and that the pop-up control could work in any program (e.g. debugger). Also Deutscher system is related to a debugger in the sense that it is a developer (author) software used for creating program presentations wherein the user can edit a program then preview, stop the preview, edit and preview again with this software (figure 11; paragraphs 137-140).

With the findings of Hampapuram, it is now even more clearly that even the combination of Deutscher and Hampapuram teaches an interface for editing source code.

A2. On page 10 of the Brief, Appellant argues that Deutscher does not teach receiving first user input hovering a mouse cursor over an expression in the source code during execution of the program; nor in response to said hovering the mouse cursor over the expression, automatically displaying a GUI element proximate to the expression, wherein the GUI element includes a value of the expression, as recited in claim 1.

R2. Examiner does not agree, these specific limitations are mentioned Deutscher and Hampapuram both teach “the user can hover with the mouse over an area of interest in a graphical user interface to display a pop-up window” singly and in combination. Hampapuram was used to cure the deficiency of how Deutscher does not disclose a debugger and source code, in which Deutscher system is open to have its control methods work for any program like a debugger having source code associated with it as taught by Hampapuram. As discussed above in the claim language rejection Deutscher describes how the user is able to interact and bring up a graphical user interface element for editing user definable information.

A3. On page 11 of Brief, Appellant argues that Deutscher and Hampapuram does not teach receiving second user input to the GUI element modifying the displayed value, thereby specifying a new value for the expression; and setting the expression in the program to the new value in response to the second user input, wherein the program continues execution in accordance with the new value of the expression, as recited in claim 1.

R3. Examiner does not agree, Deutscher directly describes displaying a value wherein the user is able to modify with the tool provided; Upon the user completing the edited changes the user is able to execute the current document that was edit to see the changes (par.137, 155, 200-201; figures 13,17 and 26B).

A4. As for claims 6, 7 and 8 on page 14 of Brief, Appellant argues that Deutscher does not teach wherein the GUI element is context sensitive.

R4. Examiner does not agree, as Deutscher provides where there are editable field where only numbers are accepted (item 1304) to program a time to when a script command is to take place (par.137,figure 13).

A5. As for claim 9, on page 15 of Brief, Appellant argues that Deutscher does not teach wherein the specified format is specified via a second GUI element in the GUI.

R5. Examiner does not agree, as Deutscher provides additional indication of what the user is to type into the editable boxes, such as the text field name “time code” and the indication of what other data looks like which is presented in the adjacent layer behind tool 1302 (time code, script type, script param.).

A6. As for claims 13-16, on pages 15-17, Appellant argues briefly that Deutscher does not teach expression that are from source code and that Hampapuram does not disclose editing source code dynamically.

R6. Examiner does not agree, for reason presented in R1 above and that Hampapuram was introduced to aid Deutscher in editing source code, further it was detailed in the analysis of the rejection that Deutscher was editing information to be presented while being able to preview the content (dynamically editing).

A7. As for claims 22 and 23 on pages 17 and 18 of Brief, Appellant argues that Deutscher does not teach evaluating the expression to determine the value of the expression.

R7. Examiner does not agree, as stated before the interface element used to edit information depicted in figure 13 at least, is shown as only being just large enough to display edit boxes and indication text.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/Nicholas Augustine/

Examiner, Art Unit 2179

Conferees:

/Ba Huynh/

Primary Examiner, Art Unit 2179

/Weilun Lo/

Supervisory Patent Examiner, Art Unit 2179

Jeffrey C. Hood
Reg. No. 35,198
Austin, TX 78767-0398
(512) 853-8800